CLICK REMOVAL IN DEGRADED AUDIO

Report for Module EEP55C22 Computational Methods

Lingyu Gong, Student ID 23337765
Trinity College Dublin
gongl@tcd.ie

October 2023

This report is submitted in part fulfilment of the assessment required in EEP55C22 Computational Methods. I have read and understand the plagiarism provisions in the General Regulations of the University Calendar for the current year. These are found in Parts II and III at http://www.tcd.ie/calendar.

This report describes the algorithm designed for the detection and removal of clicks in archived audio tracks. The sound manifests as a short sharp click or thump in the audio track this degradation arises because of noise. Our problem is to detect all the clicks and eliminate them to eliminate the noise to get clear audio.

1 Background

Audio signal degradation can be broadly categorized into two groups[2]: localized and global. Global degradation impacts the entire waveform, affecting all of its samples, and examples include background noise, hiss, flutter, and certain non-linear issues like speed variations or distortion. However, in this article, we specifically concentrate on localized degradation such as clicks, bursts, outliers, crackles, and scratches. These issues predominantly manifest in older gramophone recordings.

To restore audio signals that have been affected by sudden and impulsive noise, there are two main steps that need to be taken: detection, which involves identifying where the corrupted samples are located, and interpolation, which means replacing these flawed samples with more suitable values. Detecting the corrupted sample locations is crucial to the success of any reconstruction or interpolation method, as it heavily relies on accurately identifying where the corrupted samples are. This precision is essential to ensure unbiased parameter estimation, maintain the fidelity to the original signal, ensure iterative algorithm convergence, and more.

In cases where the degradation consists of small bursts with high amplitudes (lasting for several samples), identifying the corrupted samples is relatively straightforward, as their amplitudes significantly stand out compared to other samples. In such cases, simple techniques like using a median filter or calculating derivatives may suffice for detection. However, in most scenarios, bursts, scratches, clicks, and similar issues last from less than 20 microseconds to 4 milliseconds, which translates to a range of 1 to 200 samples (with a sampling frequency of 44.1 kHz)[2].

First, a brief introduction to the AutoRegressive Model (AR), which is a statistical model used for time series analysis and forecasting. This model assumes that there is some linear relationship between the current values of the time series data and its past values. The general form of the AR model can be expressed as below.

$$\hat{y_k} = \sum_{p=1}^{N} a_p y_{k-p} \tag{1}$$

where $\hat{y_k}$ is the predicted signal value at sample k, a_p is AR "coefficients", and y_{k-p} is the previous sample values at $k-p$. Therefore we need to divide our data into several blocks first and then estimate the coefficients by using the Least Square method, next, we are going to generate the residual and interpolate.

1.1 AR Coefficients

The coefficients, also known as parameters, in an Autoregressive Model (AR model) reflect the degree to which previous observations impact current values. These coefficients are commonly referred to as AR coefficients and are a critical aspect of the AR model[1]. They help to define the Autoregressive relationship within a time series data. The format (2) below is the solution by using the Least Square method.

$$\sum_{p=1}^{P} a_p \sum_{k=0}^{N-1} y_{k-p} y_{k-j} = \sum_{k=0}^{N-1} y_k y_{k-j} \tag{2}$$

1.2 Residual

In an AutoRegressive (AR) model, residuals refer to the differences between the predicted values of the model and the actual observed values. These residuals serve as an important tool for evaluating the quality of model fit and identifying any systematic errors present in the model[1]. Convolution methods can be applied to analyze and process residuals in AR models, which can aid in understanding periodic components and structure in time series data. The general form is shown below, assume that model order is equal to 2.

$$e_k = y_k - a_1 y_{k-1} - a_2 y_{k-2} \tag{3}$$

1.3 Interpolation

Interpolation is a technique that is commonly used to fill in gaps in data or to produce a continuous time series. It enables you to estimate values at unknown points in time so that the time series becomes continuous over the entire time horizon. The methodology and accuracy may vary depending on the specific data and problem, so you will need to choose the appropriate method based on the actual situation[1].

$$e = A_k y_k + A_u y_u \tag{4}$$

where the subscript k denotes known and the subscript u denotes unknown, and we want to get y_u such that the error is equal to 0, which leads to the following equation.

$$y_u = -\left[A_u^{'} A_u\right]^{-1} A_u^{'} A_k y_k \tag{5}$$

2 The algorithm

In my algorithm, I start by initializing parameters like the block size, duration length, and desired sampling frequency (f_s_req), and create an empty array to store the restored audio. Second step, I imported two audios $clean\_audio$ and $degraded\_audio$ to get their data, I also used both the time and frequency domains to plot their features. As the third step, the main audio restoration process is carried out in a loop for each block of audio data, I iterate through the audio in blocks of the specified size $(block\_size)$. For each block, I perform the following operations:

1. AR Coefficient Estimation: AR coefficients are estimated from the degraded block, and the mean of the block is calculated.
2. Residual Calculation: The residual of the block is calculated based on the AR model.
3. Threshold: The residual is the threshold, setting values above the threshold to zero
4. Interpolation: Missing data points are identified and interpolated using the AR model.
5. Restored blocks are concatenated to create the final restored audio.

In the final step, I save the whole audio, plot it into both time and frequency domains, and then use function$sound()$ to play, and it is easy to figure out the differences.

The algorithm focuses on using AutoRegressive (AR) modelling and interpolation techniques to restore the degraded audio signal, particularly when there are missing or corrupted data points (specifically as the flow chart below1). The AR model helps estimate and remove predictable components from the signal, while interpolation fills in missing values. The final result is a restored audio signal.

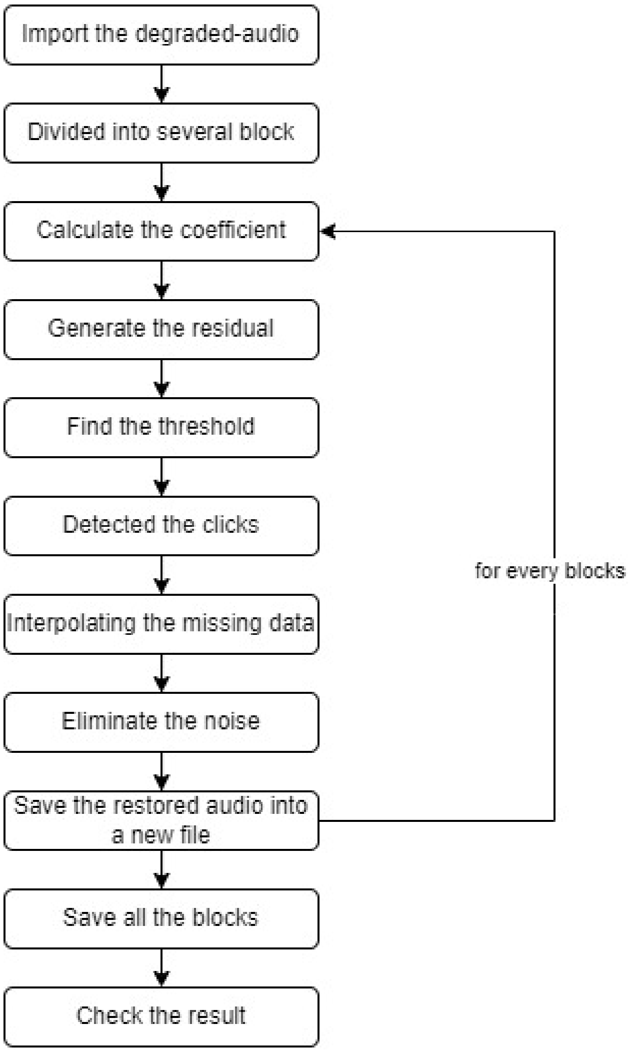

Figure 1: *The flow chart for eliminating the noise*

3 Experiments and results

My experiment entailed analyzing different audio types, namely clean, degraded, and reset audio obtained after removing noise. In the analysis process, I performed separate analyses on each audio type in the time and frequency domains. As figure 2 shows, the first line in the modified figure represents the time domain analysis while the second line represents the frequency domain analysis.

I analyzed audio signals in two ways: the time domain and the frequency domain. The amplitude of the analogue signal dropped significantly from clean to degraded audio but improved after removing the noise. The frequency spectrum of clean audio was relatively flat, while degraded audio had noise and distortion. Reset audio had a cleaner frequency spectrum, indicating a successful noise-removal process.

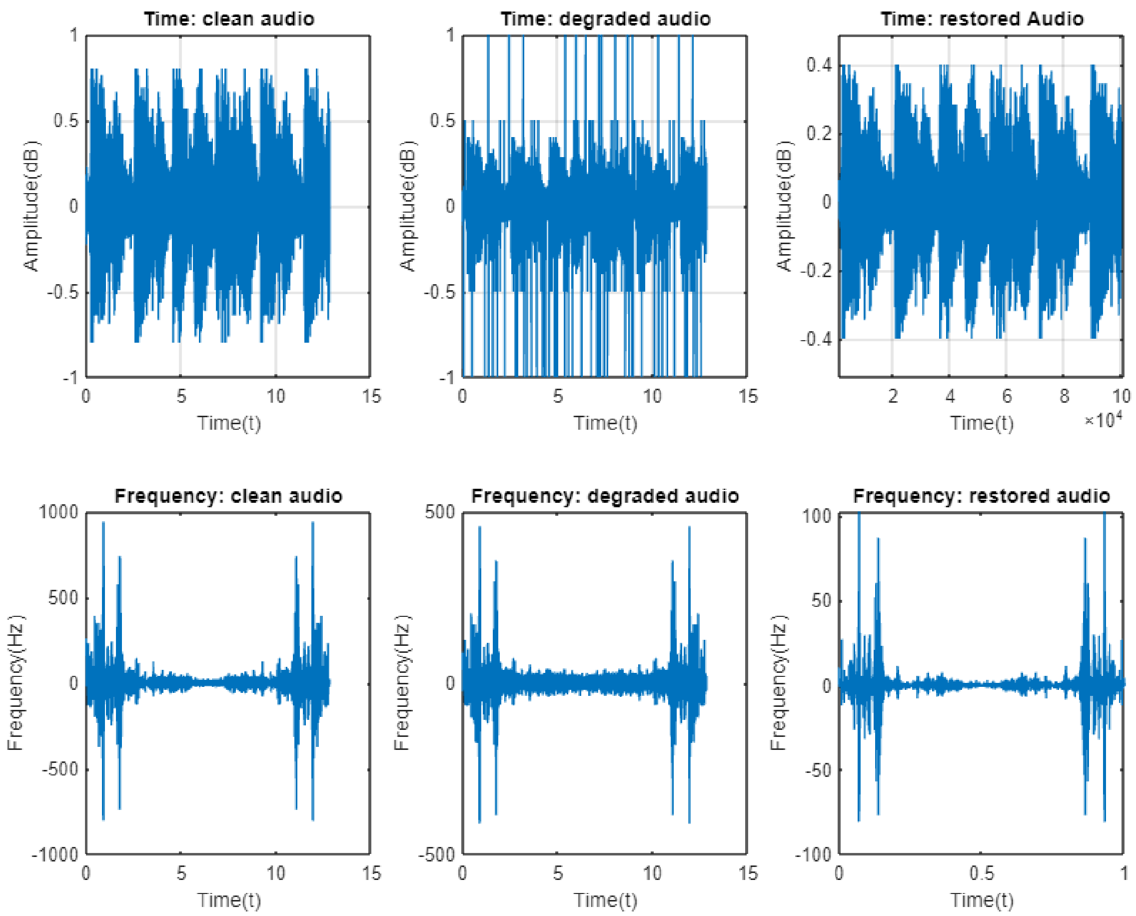

Figure 2: *Result for three types of audio*

In terms of experimental evaluation, I compared the waveform graphs before and after the audio processing, which made the differences more obvious. To measure the accuracy of the audio processing, I introduced two parameters: $detec\_num$ and $click\_num$. The former represents the number of clicks I detected, while the latter refers to the number of clicks in the original audio. A smaller difference between these two values indicates higher accuracy. I printed out the values of these parameters and plotted them as an image. For more details, please refer to Figure 3.

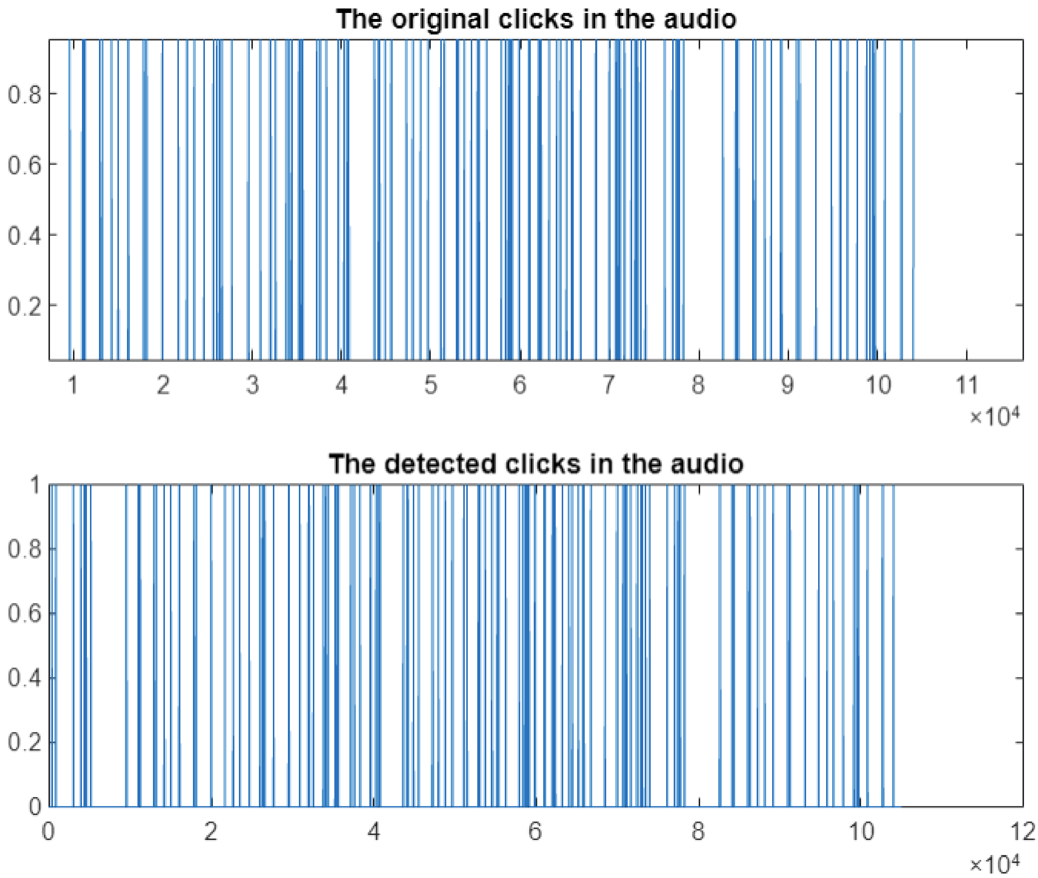

Figure 3: *Comparison of Noise Points Before and After Inspection*

4 Conclusions

During the whole project, I successfully implemented an audio restoration algorithm using AutoRegressive (AR) modelling and interpolation techniques. The algorithm takes a degraded audio signal, estimates AR coefficients, and interpolates missing or corrupted data points to restore the audio signal.

When it comes to assessing the final result, it's crucial to ensure that the evaluation process is thorough and consistent. One way to achieve this is by generating your own AI model and evaluating the results of the process by adding and removing noise yourself. This approach can provide more convincing evaluations of the current process results as it allows for a more granular examination of the results and helps to identify any potential errors or inconsistencies that might have been overlooked.

A self-generated AI model gives me greater control over the evaluation process. I can customize the model to suit your specific needs and preferences and use it to generate results that are more tailored to the requirements. This can be especially helpful when dealing with complex data sets where there might be a lot of noise or other factors that could affect the accuracy of the results.

Furthermore, generating your AI model can help to better understand how the evaluation process works. It can provide insights into the various factors that affect the results and help to identify any areas where improvements might be needed. Taking a more hands-on approach to the evaluation process can ensure that the final results are as accurate and reliable as possible.

Bibliography

[1] NIEDZWIECKI, M., AND CIOTEK, M. Elimination of clicks from archive speech signals using sparse autoregressive modeling. In *2012 Proceedings of the 20th European Signal Processing Conference (EUSIPCO)* (2012), pp. 2615–2619.

[2] OUDRE, L. Automatic Detection and Removal of Impulsive Noise in Audio Signals. *Image Processing On Line 5* (2015), 267–281. https://doi.org/10.5201/ipol.2015.64.